DOCUMENTATIE CALCULATOR DE POLINOAME

1. **Obiectivul temei:**

Obiectivul acestei teme este de a imbunatati aptitudinile programatorului in ceea ce priveste programarea pe obiecte si de a-l invata sa foloseasca un regex prin implementa un calculator de polinoame cu interfata grafica unde se pot efectua operatii precum: adunarea a doua polinoame, scaderea a doua polinoame, inmultirea a doua polinoame, impartirea a doua polinoame(cat si rest), derivarea unui polinom si integrarea unui polinom.

1. **Analiza problemei, modelare, scenarii, cazuri de utilizare:**

Se cere implementarea unui calculator de polinoame care sa efectueze operatii carecteristice polinoamelor. Deci acest calculator va avea urmatoarele functii: adunare, scadere, inmultire, impartire, derivare si integrare.

**Adunarea:**

Adunarea a doua polinoame se realizeaza adunand coeficientii termenilor celor 2 polinoame care au acelasi grad. Ex.: (2X^2+3X-1) + (3X^3-X) = 3X^3 + 2X^2 + 2X - 1. In exemplu se poate observa ca adunarea s-a facut doar intre coeficientii cu acelasi grad.

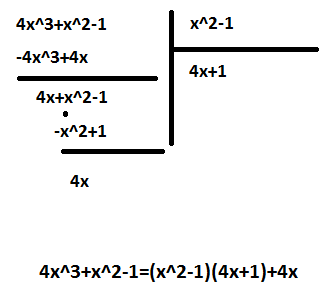
**Scaderea:**

Scaderea a doua polinoame se realizeaza precum adunarea, doar ca semnul coeficentilor celui de al doilea polinom se inverseaza. Ex.: (2X^2+3X-1) + (-3X^3+X) = -3X^3+2X^2+4X-1.

**Inmultirea:**

Inmultirea polinoamelor se realizeaza inmultind fiecare termen al primului polinom cu fiecare termen al celui de al doilea polinom. Coeficientii se inmultesc, iar gradele se aduna. Ex.: (X+1)\*(X+2) = X^2 + 2X + X+ 2 = X^2 + 3X + 2.

**Impartirea:**

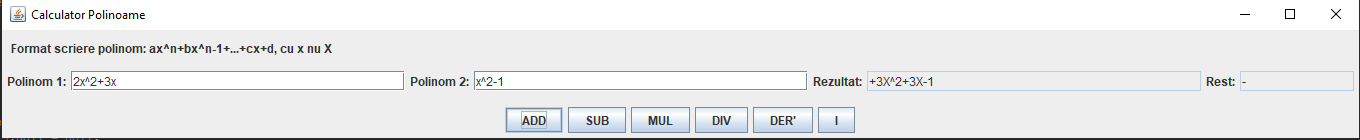
Impartirea este o operatie putin mai dificila, care are la baza un algoritm mai complex decat la celelalte operatii. Se imparte primul termen al deimpartitului la primul termen al impartitorului, acest termen se trece la cat. Termenul obtinut la impartire se inmulteste cu impartitorul, iar polinomul rezultat se scade din deimpartit, in urma scaderii obtinand un nou polinom pe care vom repta operatiile de mai sus. La finalul operatiei vom obtine un cat si un rest. Ex.:

**Derivarea:**

Derivarea unui polinom se realizeaza derivand fiecare termen al sau: coeficientul se obtine prin inmultirea coeficientului actual cu gradul actual, iar gradul se obtine scazand 1 din gradul actual. Ex.: 3X^2 + X = 6X + 1.

**Integrarea:**

Integrarea unui polinom se realizeaza integrand fiecare termen al sau: coeficientul se obtine prin impartirea coeficientului actual cu gradul actual + 1, iar gradul se obtine adunand 1 la gradul actual. Ex.: 3X^2 + X = 3/3X^3 + 1/2X^2.

 Pentru efectuarea acestor operatii trebuie sa se introduca 2 polinoame cu un format specific de la tastatura in interfata grafica, se selecteaza operatia dorita cu ajutorul butoanelor, iar rezultatul va fi afisat intr-o casuta text. In cazul unei erori de scriere a polinoamelor, utilizatorul va fi atentionat prin afisarea unui mesaj.

Cazurile de utilizare sunt diverse insa toate au un caz principal, si anume, actorul trebuie sa efectueze diverse operatii pe polinoame.

In aceste cazuri de utilizare avem un singur actor: utilizatorul.

Nu exista preconditii.

**Fluxul de evenimente obisnuit:**

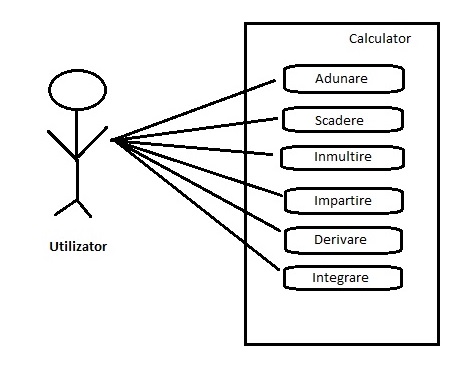
1. Utilizatorul introduce primul polinom
2. Introduce cel de al doilea polinom
3. Selecteaza operatia(Ca in exemplul de fata: adunarea)
4. In momentul apasarii butonului de operatie, se verifica daca polinoamele introduce sunt valide
5. Daca polinoamele sunt valide se afiseaza rezultatul
6. Se poate efectua o noua operatie
7. Inchidere cazului de utilizare

**Lista fluxuri alternative:**

1. Introducerea unui polinom cu format diferit decat cel specificat
   1. In acest caz se afiseaza un mesaj de eroare “Bad Input”
2. Introducerea unor coeficienti de tip double
   1. In acest caz se afiseaza un mesaj de eroare “Bad Input”
3. Primul polinom are gradul mai mic decat cel de al doilea (“doar la impartire”)
   1. In acest caz se afiseaza un mesaj de eroare, “Gradul primului polinom este mai mic”

Postconditiile acestor cazuri de utilizare:

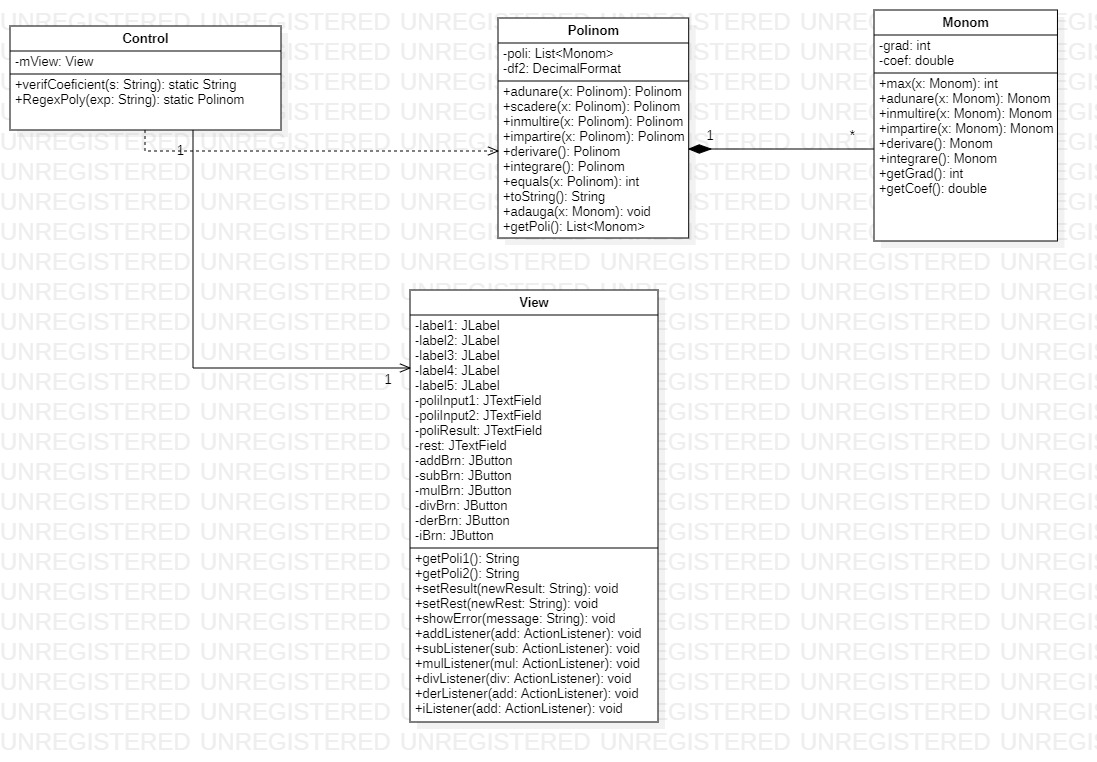
* In cazul in care datele de intrare sunt validate se afiseaza rezultatul, cazul de utilizare trece cu success
* In cazul in care datele de intrare sunt gresite se afiseaza un mesaj de eroare, cazul de utilizare se termina fara success



1. **Proiectare:**

Aplicatia contine 4 clase principale care sunt incluse in pachete separate:

1. Pachetul Model cu clasele: Polinom, Monom
2. Pachetul Control cu clasele: Control, BadInp, GradeException
3. Pachetul View cu clasa: View
4. Pachetul Main cu clasa: Main

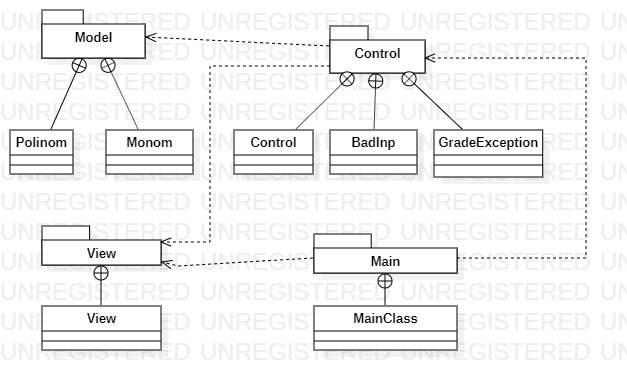
Pentru o proiectare eficienta am decis sa alcatuim polinomul dintr-o lista de monoame, astfel fiecare termen al polinomului este un monom. Acest lucru este eficient si ne ajuta in operatiile pe polinoame, deoarece toate aceste operatii au la baza operatii pe monoame. 

In imaginea de mai sus este prezentata diagrama de clase si obiecte a aplicatiei unde sunt prezentate clasele principale si legaturile dintre ele.

Intre Polinom si Monom exista o agregare puternica deoarece polinomul nu poate exista fara monoame, iar relatia este 1 la \* deoarece un polinom contine mai multe monoame.

Intre Control si Polinom avem o dependenta deoarece in clasa Control si in clasele din interiorul ei se folosesc metode cu parametri de tipul Polinom sau care returneaza un obiect de tipul Polinom.

Intre Control si View avem asociere uni-directionala deoarece, unui obiect din clasa Control ii este asociat un obiect din View, insa invers nu.

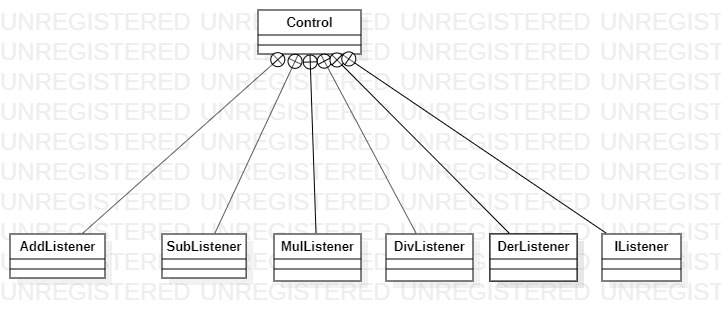


In diagrama de pachete sunt prezentate toate pachetele si clasele care fac parte din fiecare pachet in parte. Aceasta structurare pe pachete face programul mai usor de citit, dar si de facut anumite modificari, fiecare pachet avand o caracteristica aparte. De asemenea se observa si structurarea MVC, foarte eficienta cand lucrezi cu GUI( Interfata grafica). In Model se efectueaza efectiv operatiile, iar Controlul manageneriaza ceea ce trebuie sa fie afisat, cu ceea ce se intampla de fapt, avand legaturi stabilite atat cu Model cat si cu View. In View se afla clasa View care se ocupa de partea de afisare pentru utilizator, iar in Main se afla de fapt aplicatia care face toate conexiunile dintre celelalte clase si executa.

Legaturile dintre pachete pot fi traduse astfel: Control depinde de Model si de View, iar Main depinde de Control si View, ceea ce inseamna ca importa pachetele respective in cel putin una dintre clasele pachetelor lor.

Pachetul Control contine 2 clase despre care nu am discutat nimic si anume: BadInp si GradeException, ambele mostenind clasa Exception. Clasa BadInp s-a creat pentru generearea unor exceptii caracteristice unui format gresit la introducerea de la tastatura a polinoamelor. Clasa GradeException s-a creat pentru generarea unor exceptii in cazul in care gradul primului polinom este mai mare decat gradul celui de al doilea polinom. Aceasta exceptie poate fi “aruncata” doar la operatia de impartire.

De asemenea trebuie prezentat faptul ca in clasa Control avem mai multe clase interioare:



Toate aceste clase implementeaza interfata ActionListener si deservesc la asclutatorii butoanelor specifice fiecarei operatii.

In aceasta aplicatie nu se folosesc multe tipuri de structuri de date ci doar listele. Sunt foloseite in special pentru alcatuirea polinoamelor, acestea fiind de fapt o lista de monoame. S-au ales listele deoarece adaugarea este eficienta si isi pastreaza ordinea, primul element introdus fiind elemental 0 si asa mai departe.

In aceasta aplicatie nu s-au definit noi interfete, singura interfata folosita fiind cea specificata mai sus: ActionListener.

Operatiile de adunare, scadere, inmultire, derivare si integrare s-au implementat usor, nefiind nevoie de un algoritm dificil. Operatia cea mai grea a fost impartirea in care s-a folosit algoritmul de impartire intre 2 polinoame. Petru realizarea acestei operatii, se imparte primul termen al deimpartitului la primul termen al impartitorului, acest termen se trece la cat. Termenul obtinut la impartire se inmulteste cu impartitorul, iar polinomul rezultat se scade din deimpartit, in urma scaderii obtinand un nou polinom pe care vom repta operatiile de mai sus. La finalul operatiei vom obtine un cat si un rest. In capitolul 2 al documentatiei se poate gasi un exemplu.

**4.** **Implementarea:**

**Clasa Monom:**

Clasa Monom are ca variabile instanta **grad( gradul monomului) de tip int** si **coef( coeficientul) de tip double**. Metodele principale si cele mai importante ale acestei clase sunt metodele care realizeaza operatiile pe monoame, care ne sunt extrem de folositoare la operatiile pe polinoame. Fiecare metoda returneaza un Monom, care este rezultatul operatiei. De ex. avem metoda de adunare a doua monoame, primeste ca si parametru un Monom si returneaza un Monom. In cazul in care gradul celor 2 monoame sunt egale se aduna coeficientii si returneaza noul Monom, altfel metoda returneaza un Monom cu gradul 0 si coeficient 0.

**Clasa Polinom:**

Clasa Polinom are ca variabile instanta o **lista de monoame: poli**. Avem si o variabila de clasa: **df2 de tipul DecimalFormat**, pentru afisarea coeficientilor doar cu 2 zecimale. Metodele principale sunt metodele ce descriu operatiile pe polinoame, toate aceste metode returnand un Polinom.

Metoda adauga are rolul de a adauga monoame in polinom.

Metoda adunare: are ca si parametru un Polinom si returneaza un Polinom. Aceasta operatie se bazeaza pe operati de adunare dintre 2 monoame.

Atata timp cat nu s-au parcurs toti termenii celor 2 polinoame, se stabileste care polinom are gradul mai mare si in functie de asta se ia una din urmatoareale decizii:

1. daca primul polinom este mai mare se trece valoarea termenului i (monomul cu gradul cel mai mare, i-variabila contor, 0 initial) in noul polinom, iar i se incrementeaza
2. daca al doilea polinom este mai mare se trece valoarea termenului j (monomul cu gradul cel mai mare, j-variabila contor, 0 initial) in noul polinom, iar j se incrementeaza
3. in cazul in care cei doi termini au grad egal se face adunarea intre monoame, rezultatul este introdus in polinom, i si j se incrementeaza

Metoda scadere este implementata la fel ca cea pentru adunarea, cu diferenta ca semnele coeficientilor celui de al doilea polinom sunt inversate. Metoda are ca parametru un Polinom si returneaza un Polinom.

Metoda inmultire are ca si parametru un Polinom si returneaza un Polinom. Pentru implementarea operatiei se foloeseste algoritmul de inmultire a 2 polinoame, se inmulteste termen cu termen, iar la final se aduna polinoamele obtinute la polinomul current care trebuie returnat.

Metoda impartire are ca si parametru un Polinom si returneaza un Polinom. Implementarea operatiei se face conform algoritmului explicat la punctele 2 si 3.

Metodele derivare si integrare nu au parametri si returneaza un Polinom. Pentru implementare se folosesc doar operatiile de baza de la monoame.

Metoda toString este o metoda suprascrisa ce returneaza un String care reprezinta modul de afisare a polinomului pentru utilizator.

**Clasa View:**

Clasa View se ocupa cu partea vizuala a interfetei graficea, cea cu care utilizatorul intra in contact. Prin intermediul acestei clase utilizatorul poate introduce polinoamele, poate selecta butonul de operatie si de asemenea poate vizualiza rezultatul. Ca si variabile instanta contine toate etichetele, campurile text si butoanele care sunt folosite in interfata grafica. Metodele acestei clase sunt metode ce iau intrarile de la utilizator(Ex. **getPoli1**), seteaza rezultatul( Ex. **setResult**), afiseaza mesajele de eroare**( showError**) si care aplica ascultatori butoanelor( Ex. **addListener**).

**Clasa Control:**

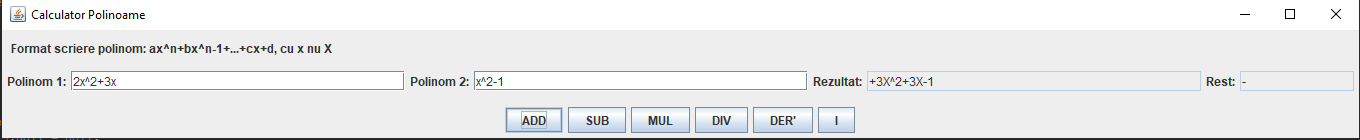
Clasa Control gestioneaza Modelul si View-ul. Are ca si variabila **instanta mView** de tip View. Astfel se pot accesa metodele din clasa View necesare aplicarii ascultatorilor pentru butoane. Contine o metoda foarte importanta si anume metoda statica **regexPoly.** Aceasta metoda are rolul de a verifica daca formatul polinoamelor introduse de catre utilizator este corect si de asemenea are rolul de a extrage coeficientii si gradele monoamelor ce alcatuiesc polinomul. Metoda arunca o exceptie de tipul BadInp.

**Clasele BadInp si GradeException:**

Sunt 2 clase ce sunt folosite pentru generarea unor exceptii. Acestea nu au nici variabile instanta, nici metode, fiind folositi doar constructorii superclasei Exception.

Interfata utilizator este implementata comform paradigmei MVC, prin care se definesc cu exactitate sarcinile fiecarei parti din program.

**Modelul** contine clasele Monom si Polinom, aici fiind implementate toate operatiiile. Implemantarile operatiilor au fost discutate mai sus.

**View-ul** contine toate elementele ce alcatuiesc interfata pe care o vede utilizatorul.

Dupa cum se poate vedea si in poza partea de aspect este implementata in View. Pentru aceasta afisare s-au folosit 5 etichete, 4 casute text, 6 butoane si 3 panouri. Doua dintre aceste panouri sunt de tipul FlowLayout(), includ cele 4 casute text cu 4 etichete si cele 6 butoane, Al treilea panou este de tip GridLayout(3,1), cu 3 linii si o coloana, unde sunt introduse cele 2 panouri anterioare si o eticheta care prezinta formatul de scriere a polinomului.

Se pot introduce date doar in primele 2 casute text, in ultimele 2 acest lucru fiind interzis prin metoda **panel.setEditable(false).**

In Control am implementat clase specifice de ascultatori pentru fiecare operatie in parte, asfel la apasarea butoanelor sa se efectueze operatia respectiva si sa se afiseze rezultatul.

**5. Rezultate:**

Pentru testarea operatiilor am folosit Junit in care am introdus diferite polinoame, iar dupa executia operatiei am comparat cu rezultatul asteptat.

Adunare:

* polinom1=4x^3+2x^2
* polinom2=2x^3
* rezultat asteptat=6x^3+2x^2

Rezultatul asteptat a fost egal cu cel obtinut in urma operatiei, testul a trecut.

Scadere:

* polinom1=4x^3+2x^2
* polinom2=2x^3
* rezultat asteptat=2x^3+2x^2

Rezultatul asteptat a fost egal cu cel obtinut in urma operatiei, testul a trecut.

Inmultire:

* polinom1=4x^3+2x^2
* polinom2=2x^3
* rezultat asteptat=8x^6+4x^5

Rezultatul asteptat a fost egal cu cel obtinut in urma operatiei, testul a trecut.

Impartire:

* polinom1=4x^3+2x^2
* polinom2=2x^3
* cat asteptat=2
* rest asteptat=2x^2

Rezultatul asteptat a fost egal cu cel obtinut in urma operatiei, testul a trecut.

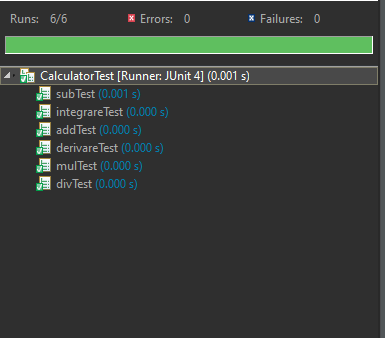
Derivare:

* polinom1=4x^3+2x^2
* rezultat asteptat=12x^2+4x^1

Rezultatul asteptat a fost egal cu cel obtinut in urma operatiei, testul a trecut.

Integrare:

* polinom1=4x^3
* rezultat asteptat=x^4

****Rezultatul asteptat a fost egal cu cel obtinut in urma operatiei, testul a trecut.

Daca se introduc toate testele prezentate mai sus cu rezultatele asteptate, in urma testului se va afisa ca in imaginea alaturata.

In cazul introducerii unui alt rezultat asteptat la oricare dintre testele de mai sus, acel test nu ar fi trecut, va aparea cu rosu.

**6. Concluzii:**

In urma acestei teme am invatat sa implementez un calculator de polinoame cu interfata grafica, sa utilizez un regex pentru anumite expresii si sa selectez anumite parti a unei expresii prin intermediul grupurilor.

Ca si dezvoltare ulterioara as implementa operatii mai grele precum gasirea radacinilor unui polinom de grad n.

**7.** **Bibliografie:**

<https://docs.oracle.com/javase/7/docs/api/index.html>

<https://regex101.com/>

<https://www.mesacc.edu/~scotz47781/mat120/notes/divide_poly/long_division/long_division.html>

[http://www.utm.mx/~caff/doc/OpenUPWeb/openup/guidances/concepts/use\_case\_BB199D1B .html?fbclid=IwAR2jDlWwwHwEQ\_V0pZN\_VIUEuvnZjHrCg9\_Ct0NrwstbOuMKrsr7DPv2kyE](http://www.utm.mx/~caff/doc/OpenUPWeb/openup/guidances/concepts/use_case_BB199D1B%20%20%20%20%20.html?fbclid=IwAR2jDlWwwHwEQ_V0pZN_VIUEuvnZjHrCg9_Ct0NrwstbOuMKrsr7DPv2kyE)